NIST Special Database 20
Scientific and Technical Document Database

Patrick J Grother
Visual Image Processing Group
Advanced Systems Division
National Institute of Standards and Technology

patrick@magi.ncsl.nist.gov

April 17, 1995

1 Introduction

The four CD ROM set, *Special Database 20* (SD20), contains the full page binary images of 23468 pages from 104 scientific and technical documents. The images were scanned from copyright-expired books provided by the NIST Library and a smaller number of technical papers donated by members of the NIST technical staff. This document details the preparation, organization and use of these images. The final database has been mastered and replicated onto four ISO-9660 formatted CD-ROM discs for permanent archiving and distribution.

The images were scanned at 15.75 dots per millimeter (400 dots per inch) and are visually of a quality comparable to the original paper pages. The images contain a very diverse range of graphic entities; halftone images, line drawings of steam turbines, maps, equations and graphs and arrays of both, multiple sizes of multicolumn machine printed text, tables, tables of contents, etc. Although the text is predominantly English, three French and German texts are included.

The contents of many of these documents were never produced using a modern typesetting package; many textual and graphic components of these images would be exceedingly difficult to recreate using contemporary software packages such as LaTeX. Thus it is the author's belief that this database contains a much richer set of challenges to document processing systems than other databases containing images synthesized from computer-borne typesetting packages. However the corollary of this is that ground truth for these images is not available, and a complete manual truthing of even a small number of certain images would be problematic and expensive. Further it is anticipated that because of the advanced technical nature of many of these documents, that complete truthing would require knowledge of the material contained therein.

Each file is accompanied by a small text file, containing basic structural information obtained from the full binary image. It includes information on size, the bounding box, the center of mass, an estimate of rotation, and blob size distribution.

None of the material has previously been published in digital format, to the best of our knowledge, and SD20 is NIST's first publication of a Scientific and Technical document database.

This document was obtained from the Postscript file sd20.cd/doc/doc.ps on Special Database 20.

partition	number of documents	number of pages	SD20 partition
st_1	27	5834	*S&T 1/4*
st_2	26	5819	*S&T 2/4*
st_3	26	5819	*S&T 3/4*
st_4	25	5996	*S&T 4/4*
total	104	23468	
st_5	25	5957	ø
total	129	29425	

Table 1: Sizes and publication statuses of the various partitions. Each partition contains distinct documents. A ø indicates that the partition has not yet been placed on CD ROM and been retained by NIST for future use.

2 Image Files *.pct

Special Database 20 has been split into four approximately equally sized partitions named *st_{1,2,3,4}*. In conjunction with this collection, NIST has prepared a similarly sized set named *st_5* which has been withheld for future use.

All documents have a unique identfier of the form *docxxx*, where the integer *xxx* runs from *001* to *129*. Thus the document entitled "Integrals of Airy Functions" is termed *doc044* and resides in partition *st_1/data* on the first CD ROM labelled as *S&T 1/4*. That document has 30 pages the rasters of which have names of the form *fxxxx.pct* where the integer *xxxx* runs from *0000* in unit increments. Each *.pct* file is accompanied by a statistics file with the *.sta* extension, the contents of which is described in section 4. An example image and its statistics file is shown in figure 1.

The image file format is discussed in detail in appendix A.

3 Index Files *.idx

One index file exists for each document. It contains three fields, the first and second identify the document and the page number while the third gives the image filename. The first field is of the form *docxxx* and the third field gives the image filename. Both of these formats are described above. The second field is of the form *pppp.nnn* where *pppp* is the actual page number keyed from the document. The "subpage" identifier *nnn* is usually *000*, but for pages with fold out inserts or merely subscripted page numbers it is a counter of sequential pages. Thus a page number sequence *41, 42-1, 42-2, 43* will be reported in the index file as *0041.000, 0042.000, 0042.001, 0043.000*. Many of the books start with maybe one entirely unnumbered page and then a Roman numeral run, *i, ii, iii* before the first numbered page. The corresponding *.idx* head looks like this:

```
doc006      0000.000 f0000.pct
doc006      0000.001 f0001.pct
doc006      0000.002 f0002.pct
doc006      0000.003 f0003.pct
doc006      0001.000 f0004.pct
doc006      0002.000 f0005.pct
```

In cases where the page was blank except for the page numbering there is no corresponding image, nor *.idx* entry. Generally the image *.pct* filenames **do not** correspond to the page numbers and the second field of the *.idx* file does not generally yield the actual page text contained in the image. The problems associated with fold out inserts, blank page deletions, and subscripted page numbers, make representative filenaming difficult.

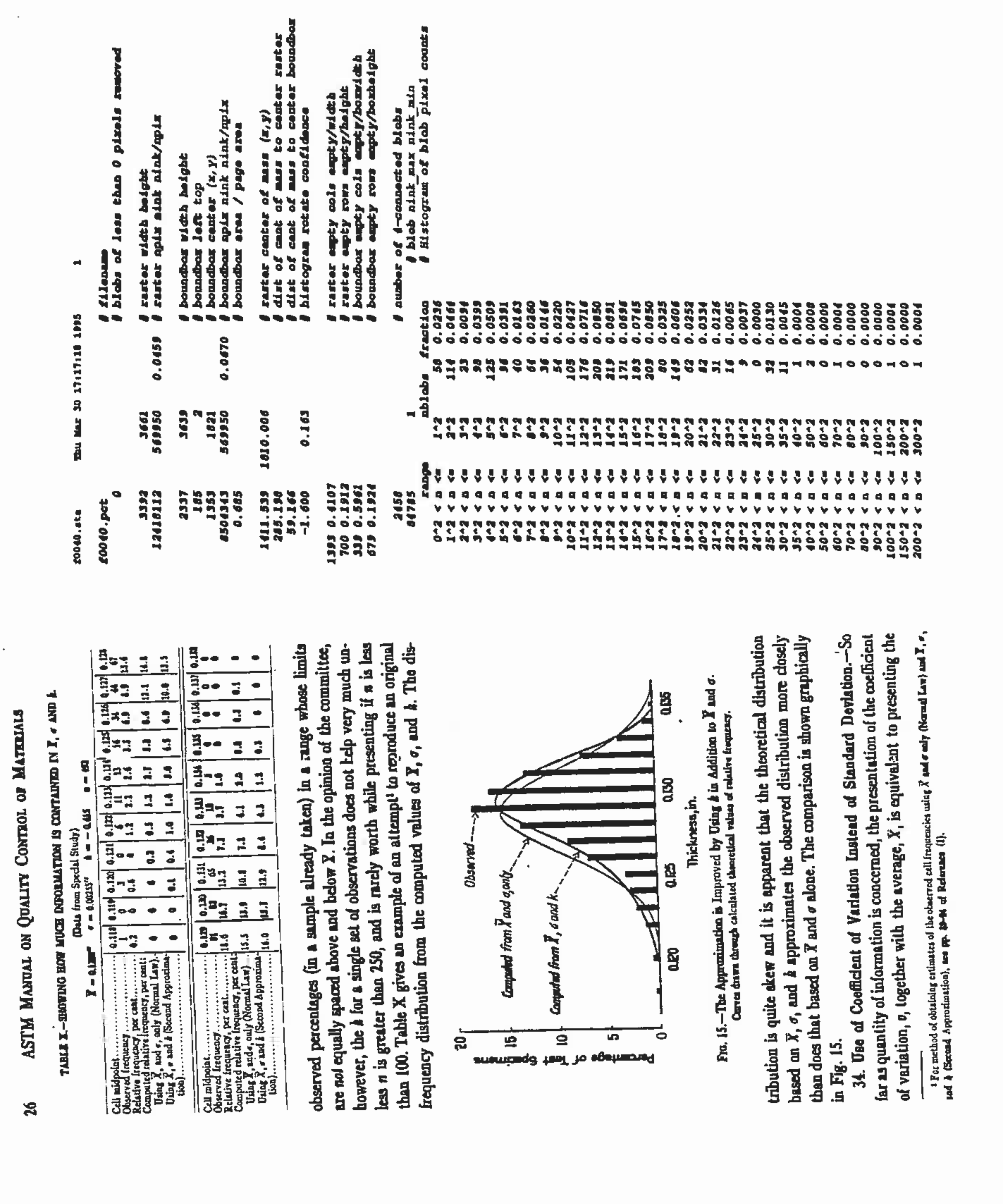

26 ASTM MANUAL ON QUALITY CONTROL OF MATERIALS

TABLE X.—SHOWING HOW MUCH INFORMATION IS CONTAINED IN $\bar{X}$, σ AND k.

(Data from Special Study)

$\bar{X}$ = 0.12[illegible]″ σ = 0.00255″ k = −0.415 n = 4[illegible]

Cell midpoint	0.118	0.119	0.120	0.121	0.122	0.123	0.124	0.125	0.126	0.127	0.128
Observed frequency	1	0	3	0	6	11	13	16	34	44	67
Relative frequency, per cent	0.2	0	0.6	0	1.2	2.2	2.6	3.2	6.9	8.9	13.6
Computed relative frequency, per cent: Using $\bar{X}$, and σ, only (Normal Law)	0	0	0	0.3	0.5	1.2	2.7	5.3	8.6	12.1	14.8
Using $\bar{X}$, σ and k (Second Approximation)	0	0	0.1	0.4	1.0	1.6	3.0	4.5	6.9	10.0	13.5
Cell midpoint	0.129	0.130	0.131	0.132	0.133	0.134	0.135	0.136	0.137	0.138	
Observed frequency	91	83	65	36	18	5	0	0	0	0	
Relative frequency, per cent	18.6	16.7	13.2	7.3	3.7	1.0	0	0	0	0	
Computed relative frequency, per cent: Using $\bar{X}$, and σ, only (Normal Law)	15.5	13.9	10.8	7.3	4.1	2.0	0.8	0.3	0.1	0	
Using $\bar{X}$, σ and k (Second Approximation)	16.0	15.7	12.9	8.4	4.3	1.2	0.3	0	0	0	

observed percentages (in a sample already taken) in a range whose limits are *not* equally spaced above and below $\bar{X}$. In the opinion of the committee, however, the k for a single set of observations does not help very much unless n is greater than 250, and is rarely worth while presenting if n is less than 100. Table X gives an example of an attempt[1] to reproduce an original frequency distribution from the computed values of $\bar{X}$, σ, and k. The dis-

FIG. 15.—The Approximation is Improved by Using k in Addition to $\bar{X}$ and σ. Curves drawn through calculated theoretical values of relative frequency.

tribution is quite skew and it is apparent that the theoretical distribution based on $\bar{X}$, σ, and k approximates the observed distribution more closely than does that based on $\bar{X}$ and σ alone. The comparison is shown graphically in Fig. 15.

34. Use of Coefficient of Variation Instead of Standard Deviation.—So far as quantity of information is concerned, the presentation of the coefficient of variation, v, together with the average, $\bar{X}$, is equivalent to presenting the

[1] For method of obtaining estimates of the observed cell frequencies using $\bar{X}$ and σ only (Normal Law) and $\bar{X}$, σ, and k (Second Approximation), see pp. 80–84 of Reference (1).

```
f0040.sta         Thu Mar 30 17:17:18 1995        1

f0040.pct                                   # filename
        0                                   # blobs of less than 0 pixels removed

     3392      3661                         # raster width height
 12418112    569950     0.0459              # raster npix nink nink/npix

     2337      3639                         # boundbox width height
      185         2                         # boundbox left top
     1353      1821                         # boundbox center (x,y)
  8504343    569950     0.0670              # boundbox npix nink nink/npix
    0.685                                   # boundbox area / page area

 1411.539  1810.006                         # raster center of mass (x,y)
  285.198                                   # dist of cent of mass to center raster
   59.166                                   # dist of cent of mass to center boundbox
   -1.600     0.163                         # histogram rotate confidence

1393 0.4107                                 # raster empty cols empty/width
 700 0.1912                                 # raster empty rows empty/height
 339 0.5961                                 # boundbox empty cols empty/boxwidth
 679 0.1924                                 # boundbox empty rows empty/boxheight

     2458                                   # number of 4-connected blobs
    84785         1                                   # blob nink_max nink_min
        range        nblobs  fraction                 # Histogram of blob pixel counts
  0^2 < n <=   1^2     58  0.0236
  1^2 < n <=   2^2    114  0.0464
  2^2 < n <=   3^2     23  0.0094
  3^2 < n <=   4^2     98  0.0399
  4^2 < n <=   5^2    125  0.0509
  5^2 < n <=   6^2     96  0.0391
  6^2 < n <=   7^2     40  0.0163
  7^2 < n <=   8^2     64  0.0260
  8^2 < n <=   9^2     36  0.0146
  9^2 < n <=  10^2     54  0.0220
 10^2 < n <=  11^2    105  0.0427
 11^2 < n <=  12^2    176  0.0716
 12^2 < n <=  13^2    208  0.0850
 13^2 < n <=  14^2    219  0.0891
 14^2 < n <=  15^2    171  0.0696
 15^2 < n <=  16^2    183  0.0745
 16^2 < n <=  17^2    209  0.0850
 17^2 < n <=  18^2     80  0.0325
 18^2 < n <=  19^2    149  0.0606
 19^2 < n <=  20^2     62  0.0252
 20^2 < n <=  21^2     82  0.0334
 21^2 < n <=  22^2     31  0.0126
 22^2 < n <=  23^2     16  0.0065
 23^2 < n <=  24^2      9  0.0037
 24^2 < n <=  25^2      0  0.0000
 25^2 < n <=  30^2     32  0.0130
 30^2 < n <=  35^2     11  0.0045
 35^2 < n <=  40^2      1  0.0004
 40^2 < n <=  50^2      2  0.0008
 50^2 < n <=  60^2      0  0.0000
 60^2 < n <=  70^2      1  0.0004
 70^2 < n <=  80^2      0  0.0000
 80^2 < n <=  90^2      0  0.0000
 90^2 < n <= 100^2      0  0.0000
100^2 < n <= 150^2      1  0.0004
150^2 < n <= 200^2      0  0.0000
200^2 < n <= 300^2      1  0.0004
```

Figure 1: Example S & T Image. At bottom is the image of file *st_1/data/doc010/f0040.pct*, which is rotated by -1.6 degrees, relative to the horizontal datum line. The negative value indicates counter-clockwise rotation. This value is noted on line 16 of the corresponding statistics file *f0040.sta*, shown at top.

4 Statistics Files *.sta

An example statistics file is given alongside its image in figure 1. The entries are followed by '#' delimited comments. All *.sta* files were obtained using the *imgstats* executable whose C source code is found in the directory st_?/src/bin/imgstats. The first line names the input image file. The second line states an integer four-way connected component filtering threshold; the statistics contained in the remainder of the file were obtained from the input image minus any blobs whose size is *less* than this number of pixels. Lines four and five give the whole image width, and height, its area, the number of black ink pixels, and a global ink density. The next five lines contain the top left hand coordinates of the bounding box, its width height and absolute center, its area, number of black pixels, the (increased) density, and the ratio of bounding box to page areas.

This data is followed by the absolute coordinates of the center of mass i.e. the mean x and y values of the black pixels, the distances of the center of mass to the centers of the image and the bounding box, and finally an estimate of the rotation of the text and graphics relative to the coordinate axes. This angle is given in degrees, and is accompanied by a confidence measure between zero and one. Appendix B gives a full description of the orientation estimation method.

The next four lines give the numbers of completely blank columns and rows in the raster, then in the bounding box. These numbers are also expressed as fractions of their widths and heights.

The final section contains the number of distinct blobs and the number of pixels contained in the largest and smallest blobs. The table follows giving a histogram of blob sizes, where the t^{th} entry gives the number of connected components found in the image whose number of pixels n lies between $(t-1)^2+1$ and t^2.

Note that the bounding box and blob statistics are very sensitive to the value of blob size threshold filtering parameter supplied optionally to *imgstats*. The default value, used on all images of this CD ROM, is 1, meaning that no blobs were removed prior to processing.

5 Data Hierarchy

All CD ROMs of the four disc volume have identical directory hierarchies. Between each partition differences exist only in the *data* (image) and *misc* (auxilliary image information) trees.

```
st_1  st_2  st_3  st_4    ( st_5 )
|
bin     data     doc     include     lib     makefile.mak     misc     src
|       |        |       |           |                        |        |
```

The *bin* and *lib* directories are the destinations for executables and libraries whose source is held in the *src/bin/** and *src/lib/** trees. The *include* directory contains C header files pertinent to that code. The file *makefile.mak* initiates a recursive compilation of the source and is described further in section 7. The documentation including the LaTeX and Postscript for this report are contained in the *doc* directory. The *misc* tree contains full CD listings, sizes, titles, and other information associated with the preparation of this database. Finally the *data* tree contains the images organized by document thus:

```
st_4
|
|
data
|
|
doc001  doc003  doc004 ... doc076  doc077
```

```
|
|
f0000.pct f0001.pct ... f0013.pct  doc001.idx
f0000.sta f0001.sta ... f0013.sta
```

The *misc* tree contains several extra files. The *.prt* file shows which documents are contained on this partition and the *.ttl* file contains the titles of those documents. The *.ls* file contains a UNIX recursive long listing of the *st_?* hierarchy, and the *.siz* file gives the total integer sum of the byte sizes of the compressed image *.pct* files.

```
st_4
|
|
misc
|
|
st_4.ttl st_4.prt st_4.ls st_4.siz
```

6 Caveats

1. Many operating systems and shells have upper limits on the number of arguments that may be supplied on the command line to installed programs. Users of the largest documents *st_3/doc056* and *st_1/doc023* will find wildcard expansions to be time-consuming. The problem is exacerbated by the slow access times typical of CD ROM's.

2. A very small number of the images were scanned from the dark covers of pamphlet-like documents, and are very noisy. The files are typically very large and underexposed, and contain a small amount of title text. Note that the compression actually increases the data size of very noisy images, by up to a theoretical maximum of a factor of 2.5, and as a result a small number of images exceed a megabyte in size.

3. If a user possesses NIST Special Databases published prior to SD19 (March 1995) and employs the supplied CCITT Group 4 compression code then it should be updated with this SD20 release. This modified code allows very large image files to be handled. Use of the old compression version on these files **will result in the loss of all the data.**

7 Software Utilities

A number of C coded utilities have been included in this CD firstly to provide tools for handling the provided image data, and secondly to give examples of how to handle the NIST IHEAD structure. Because the CD ROM is read-only storage the *src* and *include* directories need to be copied to a writable disk prior to compilation. In addition directories named *bin* and *lib* must be created. Given a copy of these trees from say */cd* to, for example, the directory */usr/local/sd20*, the executable binaries for various architectures can be produced in the *bin* directory by invoking the UNIX commands thus:

```
# mkdir /usr/local/sd20
# cd /usr/local/sd20
# cp -rp /cd/src /cd/include /cd/man /cd/makefile.mak .
# mkdir bin lib
# make -f makefile.mak instarch PROJDIR=/usr/local/sd20 INSTARCH=sgi
# make -f makefile.mak depend PROJDIR=/usr/local/sd20
# make -f makefile.mak install PROJDIR=/usr/local/sd20
```

where the environment variable PROJDIR is the root path to the *src* directory, and INSTARCH, which indicates the desired machine architecture, is one of *sun*, *sgi*, *aix*, *hp*, *sol* or *osf*. A brief description of the software utilities is given below. The manual pages that follow offer a more complete description.

imgstats
This utility takes a number of IHEAD images producing a statistics file for each.

dumpihdr
This utility dumps the textual information of the header of a NIST ihead file to standard output.

decomp
Removes IHEAD supported CCITT Group IV compression and overwrites the input image file.

compgrp4
Applies Group IV compression and overwrites the input IHEAD file.

References

[1] Michael D. Garris. Design, Collection, and Analysis of Handwriting Sample Image Databases. *The Encyclopedia of Computer Science and Technology*, 31:189–213, 1994.

Appendix A - IHEAD Image Format

NIST's Visual Image Processing Group has developed the IHEAD image header and has incorporated it into all images of its Special Databases. The IHEAD design and functionality are described extensively in [1]. All members of the header are ASCII coded with some fields being in 'C' string format and other fields being single ASCII characters. The actual structure definition for an NIST header and a description of the use and purpose of each member is shown below. The header is of a fixed allocated length, currently 296 bytes, regardless of its content. The first eight bytes contain the length of the remainder, i.e. 288 bytes all of which can be read into structure memory with a single "fread" system call.

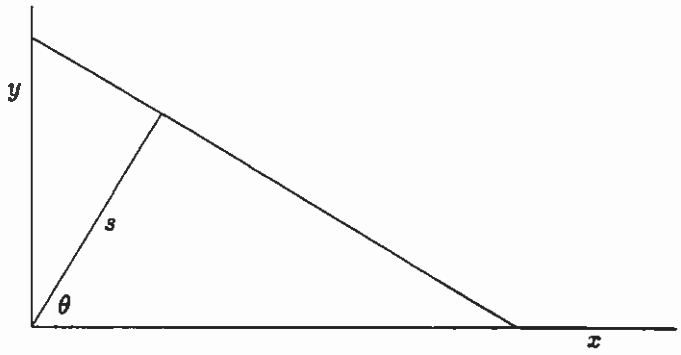
All the SD20 images have been compressed using an implementation of the CCITT Group 4 algorithm developed by the CALS Test Network and adapted by NIST for use with files which have compressed images and uncompressed headers. The IHEAD header within each image remains uncompressed with the appropriate members set to reflect the fact that the following raster data have been compressed before being saved to a file. The file *src/lib/image/readrast.c* contains the C routine *ReadBinaryRaster* that when given an IHEAD image file will allocate, load, and return the header structure with the decompressed raster data. Although an IHEAD file may contain uncompressed data, the reading routine will invoke, if necessary, routines to decompress the image data. The files *src/lib/image/grp4{comp,deco}.c* contain modified source code for compressing and decompressing binary rasters.

```
/* Defines used by the ihead structure */
#define IHDR_SIZE       288     /* len of hdr record (always even bytes) */
#define SHORT_CHARS     8       /* # of ASCII chars to represent a short */
#define BUFSIZE         80      /* default buffer size */
#define DATELEN         26      /* character length of date string */

typedef struct ihead{
   char id[BUFSIZE];                    /* identification/comment field */
   char created[DATELEN];               /* date created */
   char width[SHORT_CHARS];             /* pixel width of image */
   char height[SHORT_CHARS];            /* pixel height of image */
   char depth[SHORT_CHARS];             /* bits per pixel */
   char density[SHORT_CHARS];           /* pixels per inch */
   char compress[SHORT_CHARS];          /* compression code */
   char complen[SHORT_CHARS];           /* compressed data length */
   char align[SHORT_CHARS];             /* scanline multiple: 8|16|32 */
   char unitsize[SHORT_CHARS];          /* bit size of image memory units */
   char sigbit;                         /* 0->sigbit first | 1->sigbit last */
   char byte_order;                     /* 0->highlow | 1->lowhigh*/
   char pix_offset[SHORT_CHARS];        /* pixel column offset */
   char whitepix[SHORT_CHARS];          /* intensity of white pixel */
   char issigned;                       /* 0->unsigned data | 1->signed data */
   char rm_cm;                          /* 0->row maj | 1->column maj */
   char tb_bt;                          /* 0->top2bottom | 1->bottom2top */
   char lr_rl;                          /* 0->left2right | 1->right2left */
   char parent[BUFSIZE];                /* parent image file */
   char par_x[SHORT_CHARS];             /* from x pixel in parent */
   char par_y[SHORT_CHARS];             /* from y pixel in parent */
}IHEAD;
```

```
   1. id          - General identification field used to contain the
                        image file name and any other information useful for
                        image distinction.
   2. created     - Date when the image was created.
```

3. width - Pixel width of the image.
4. height - Number of scanlines in the image.
5. depth - Number of bits representing a single pixel.
6. density - Pixels per inch resolution of the image.
7. compress - ASCII code used to represent the compression algorithm used on the image.
8. complen - Length of compressed image in bytes.
9. align - Even multiple in bits to which each scanline is padded.
10. unitsize - Size in bits of fundamental data units in the image.
11. sigbit - Order of most significant to least significant bits within fundamental data units in the image.
12. byte_order - Order of high and low bytes used if fundamental data units are 2 bytes long in the image.
13. pix_offset - Pixel offset into the data where the image of interest actually begins.
14. whitepix - Intensity value of white pixels in the image.
15. issigned - Flag which signifies whether fundamental data units include a sign bit or are unsigned.
16. rm_cm - Flag which signifies whether the raster data is stored row-major or column-major.
17. tb_bt - Flag which signifies whether the raster data is stored top-to-bottom or bottom-to-top.
18. lr_rl - Flag which signifies whether the raster data is stored left-to-right or right-to-left.
19. parent - File name of the parent image if the image is a subimage.
20. par_x - X coordinate pixel value in the parent image where the subimage was cut.
21. par_y - Y coordinate pixel value in the parent image where the subimage was cut.



Appendix B - Rotation Detection

The method employed in SD20 for estimating the angle of rotation given in the *.sta* files is, in our experience, quite robust, though, unfortunately, not particularly efficient. Although a theoretical description follows, the user is directed to the source code in *st_?/src/bin/imgstats/deskew.c* for the straightforward implementation.

Consider a line in the (x, y) plane. The line from the origin that intersects this line perpendicularly is inclined at angle θ as shown below. Consider a line that is perpendicularly intersected by a line from the origin whose length is s that makes an angle θ with the horizontal. Any point (x, y) that lies on the first line obeys

$$y = -x \tan(\pi/2 - \theta) + s/\sin\theta$$

$$s = x \sin\theta + y \cos\theta$$

where (s, θ) is known as the Hough transform of the line. So given an image, $I(x, y)$, and adopting this notation we define the Radon transform to be the line integral of the image along that line defined by (s, θ).

$$R(s, \theta) = \int_y \int_x I(x, y)\ \delta(s - x\sin\theta - y\cos\theta)\ dxdy$$

where $-\infty < s < \infty$, $0 \leq \theta < \pi$, and the δ function is unity only if its argument is zero, corresponding to the point lying on the line. The left hand side, $R(s, \theta)$, is known as the ray sum. For the purposes of orientation detection we define the angular power as the integral over all displacements of the squared Radon transforms of parallel rays:

$$\nu(\theta) = \int_s R(s, \theta)^\beta\ ds$$

where we have used $\beta = 2$ though any value greater than unity would be suitable. Thus it is clear that $\nu(\theta)$ will have a large value if there is significant structure in the image that is directed at that particular angle. For a sampled discrete image the ray sum is calculated as the sum of the pixel values along the line defined by (s, θ). In the case of binary images we take $I(x, y)$ to be unity only if there is ink at location (x, y). Otherwise it is zero, and the power quantity is just the sum of the squares of the number of pixels lying on the inclined ray vectors. Given an image of width w and height h the quantity becomes

$$\nu(\theta) = \sum_{s=1}^{h} \left(\sum_{j=1}^{w} I(x_j, y_j) \right)^\beta$$

where instead of using the analytic formula, $y_j = s\csc\theta - x_j \cot\theta$, the implementation actually constructs ray vectors using Bresenham's more efficient line drawing algorithm. Thus for discrete values of θ on the range $-6 \leq \theta \leq 6$ with an increment of 0.2 degrees, the method accumulates, for each vertical displacement (i.e. row index, s), the squared sum of the number of dark image pixels lying on the line yielding a power measure as a function of θ as shown in figure 2.

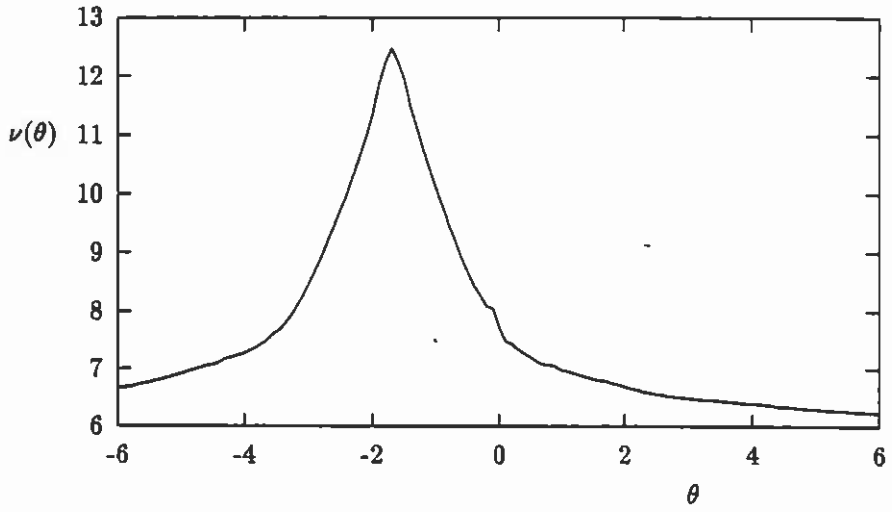


Figure 2: The sum of the squared vector sums of the image *st_1/data/doc010/f0040.pct* as a function of angle. The angle for which ν is maximum corresponds to the rotation of the page.

NAME

compgrp4 - overwrites a file with its data Group IV compressed.

SYNOPSIS

compgrp4 input.pct

DESCRIPTION

compgrp4 is a utility that takes a NIST IHEAD image file, compresses the data using CCITT Group IV and outputs a typically smaller NIST IHEAD file.

OPTIONS

EXAMPLES

Given the input file /tmp/f0358.pct its data may be compressed thus:

```
compgrp4  /tmp/f0358.pct
```

FILES

st_1/include/ihead.h
st_1/include/grp4comp.h
st_1/src/lib/image/grp4comp.c

SEE ALSO

decomp

BUGS

A feature of the Group IV compression is that noisy decorrelated data streams may actually increase in size under "compression". The maximum increase is a factor of 5/2.

Users of NIST Image Recognition Group supplied Group IV compression released before February 1995 should update their installed software. The code has been recently modified to allow very tall files to be handled correctly; previous releases have severely corrupted such files.

NAME

decomp - writes file of decompressed image data

SYNOPSIS

decomp [-o output.pct] input.pct ...

DESCRIPTION

decomp is a utility that takes a NIST IHEAD image file, decompresses the CCITT_IV data and outputs a typically much larger NIST IHEAD file.

OPTIONS -o output.pct Without this option the file "input.pct" is overwritten. The -o flag allows the user to specify an alternative output path and filename.

EXAMPLES

Given the st_1 data hierarchy from the CD ROM, the file doc005/f0358.pct can be decompressed and output to /tmp as follows:

decomp -o /tmp/f0358.pct /sd20/st_1/data/doc005/f0358.pct

FILES

st_1/include/ihead.h
st_1/include/grp4deco.h
st_1/src/lib/image/grp4deco.c

SEE ALSO

compgrp4

BUGS

NAME

dumpihdr - prints one or more image file's IHEAD header information.

SYNOPSIS

dumpihdr input.pct ...

DESCRIPTION

dumpihdr is a utility that dumps image header information to standard output for multiple NIST IHEAD format image files.

OPTIONS

EXAMPLES

Given the SD20 data hierarchy, the IHEAD structures of the first writer's completed HSF and digit segmentations are written to standard output using:

```
dumpihdr st_1/data/doc010/f0040.pct
```

FILES

st_1/include/ihead.h
st_1/include/mis.h

SEE ALSO

BUGS

NAME

imgstats - produce statistics from an image

SYNOPSIS

imgstats [-rb blobsize] input1.pct ...

DESCRIPTION

imgstats is a utility that takes IHEAD *.pct* files, produces a statistics *.sta* file for each input image.

OPTIONS

-r do not estimate the text rotations, with consderable time savings.

-b blobsize before obtaining statistics remove any blobs smaller than this integer number of pixels. The input image files are not corrrupted.

EXAMPLES

Given the first SD20 data hierarchy, st_1, the statistics of the file shown in figure 1 are computed thus:

```
imgstats st_1/data/doc010/f0040.pct
```

The *.sta* file will be overwritten if we decide to recompute the statistics while ignoring single pixel blobs thus:

```
imgstats -rb2 st_1/data/doc010/f0040.pct
```

FILES

st_1/include/ihead.h
st_1/include/image.h
st_1/src/bin/imgstats/imgstats.h

SEE ALSO

BUGS

The output file is overwritten if the executable is rerun on the same input image.

The rotation estimate is too expensive. It is more economical to reduce the image size (zoom down) and run imgstats on the smaller result.